



XAPP1023 (v1.0) October 3, 2007

Benchmarking the Performance of the Virtex-4 10/100/1000 TEMAC System

Author: Kris Chaplin

Abstract

This application note provides step-by-step instructions on how to recreate a Tri-Mode Ethernet (TEMAC) performance testing system using the ML405 board and MontaVista Linux 4.0. This application note shows how to set up a simple EDK Base System Builder system on the ML405 Evaluation Platform and run performance tests. The network architecture for the test is described. A system is built and downloaded into the FPGA. A MontaVista Linux kernel is configured, built, and downloaded into the ML405 Evaluation Platform. The instructions for obtaining and setting up the software used to perform the measurements, netperf, are given.

Included Systems

Included with this application note is one reference system, ml405_ppc_plb_temac built for the Xilinx ML405 Rev A board. The reference system is available at:

www.xilinx.com/bvdocs/apnotes/xapp1023.zip

Hardware and Software Requirements

The hardware and software requirements for this reference system are:

- Xilinx ML405 Evaluation Platform
- Xilinx 9.1 EDK and ISE™ tools
- A Linux machine with root access and a network interface capable of supporting gigabit Ethernet and jumbo frames (this test uses the Intel 82540EM chipset)
- A CAT5e network patch cable

Test Infrastructure Setup

- A desktop PC running a supported operating system, with XPS and EDK 9.1 with their latest service packs
- A linux PC running a version of Linux, with an exported root file system for the linux target
- An ML405 demonstration board, with debug ports (RS232 and JTAG) connected to the desktop PC, and a CAT5e network cable attached to the linux PC.

The desktop PC and Linux PC could be the same machine, if running a supported operating system. For the purpose of performance and simplicity in this test setup, two machines were used.

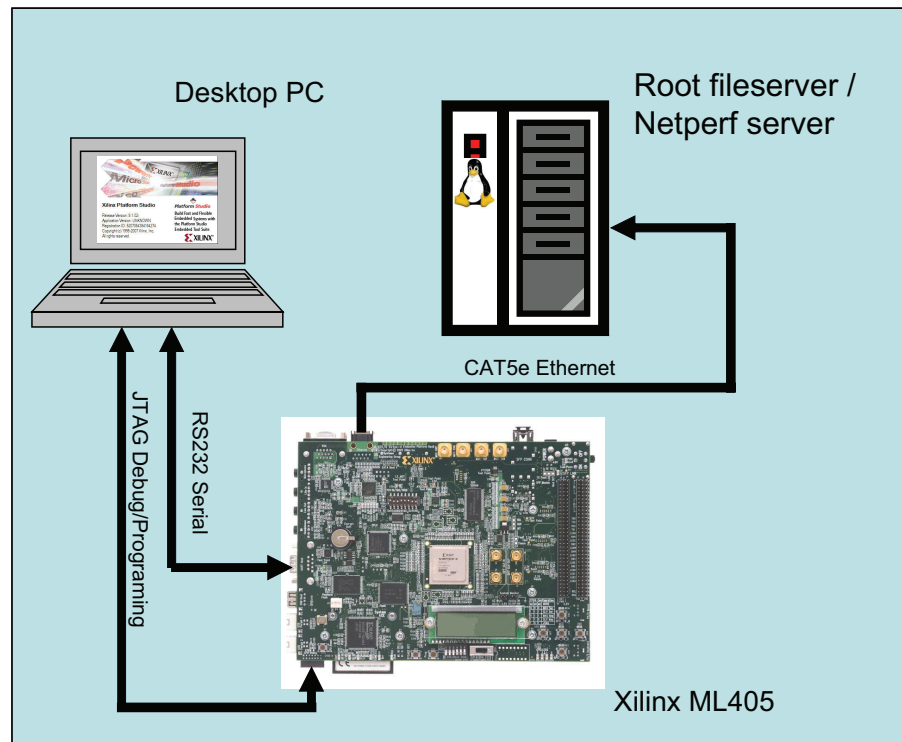
© 2007 Xilinx, Inc. All rights reserved. All Xilinx trademarks, registered trademarks, patents, and further disclaimers are as listed at <http://www.xilinx.com/legal.htm>. PowerPC is a trademark of IBM Inc. All other trademarks and registered trademarks are the property of their respective owners. All specifications are subject to change without notice.

NOTICE OF DISCLAIMER: Xilinx is providing this design, code, or information "as is." By providing the design, code, or information as one possible implementation of this feature, application, or standard, Xilinx makes no representation that this implementation is free from any claims of infringement. You are responsible for obtaining any rights you may require for your implementation. Xilinx expressly disclaims any warranty whatsoever with respect to the adequacy of the implementation, including but not limited to any warranties or representations that this implementation is free from claims of infringement and any implied warranties of merchantability or fitness for a particular purpose.

Reference System Specifics

This system uses the PPC405, OPB UART 16550, and PLB TEMAC

See [Figure 1](#) for the interconnect diagram of the system used in the performance tests.

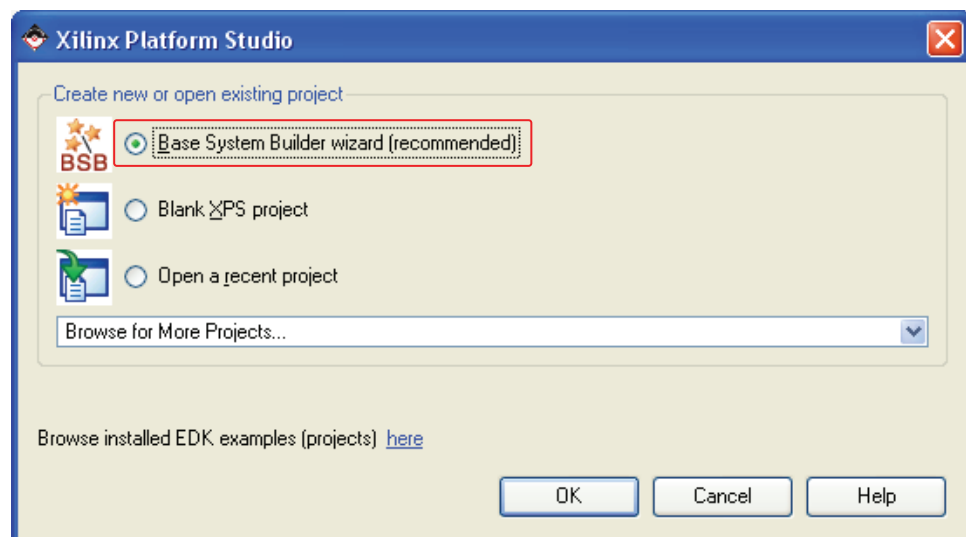


X1023_01_091207

Figure 1: Interconnect Diagram

Generating the Test Hardware System

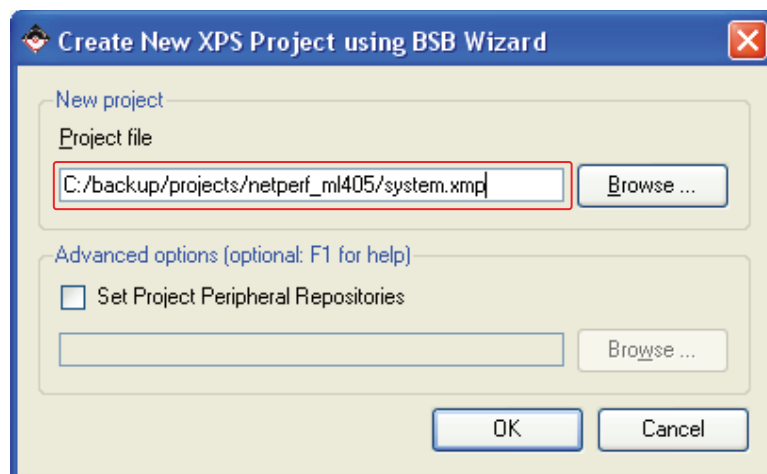
The base system builder (BSB) is used to generate a basic processor architecture, which is then optimized for performance later. Invoke **File** → **New project** to start the base system builder from XPS 9.1. In the XPS window shown in [Figure 2](#), select **Base System Builder Wizard**, then click **OK**.



X1023_02_091207

Figure 2: Starting the Base Buidler System (BSB)

In the BSB window shown in Figure 3, select a directory for the new project, then click **OK** to continue.



X1023_03_091207

Figure 3: Creating a New Project Directory

In the BSB - Welcome window shown in Figure 4, select **I would like to create a new design**, then click **Next** to create a new design.



X1023_04_091207

Figure 4: Welcome to BSB

In the BSB - Select Board window shown in [Figure 5](#), as Board vendor, select **Xilinx**, select **Virtex 4 Xilinx ML405 Evaluation Platform** as the Board name, then click **Next**.

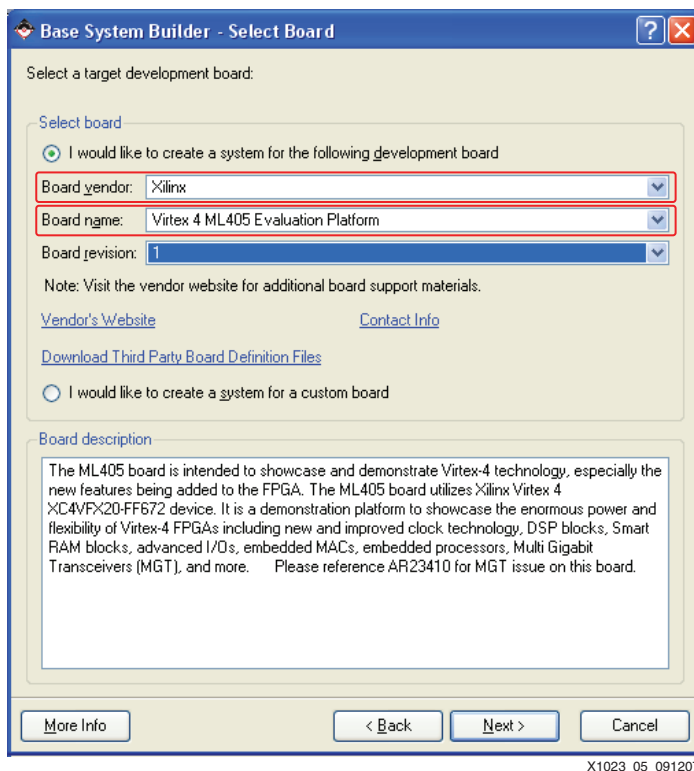
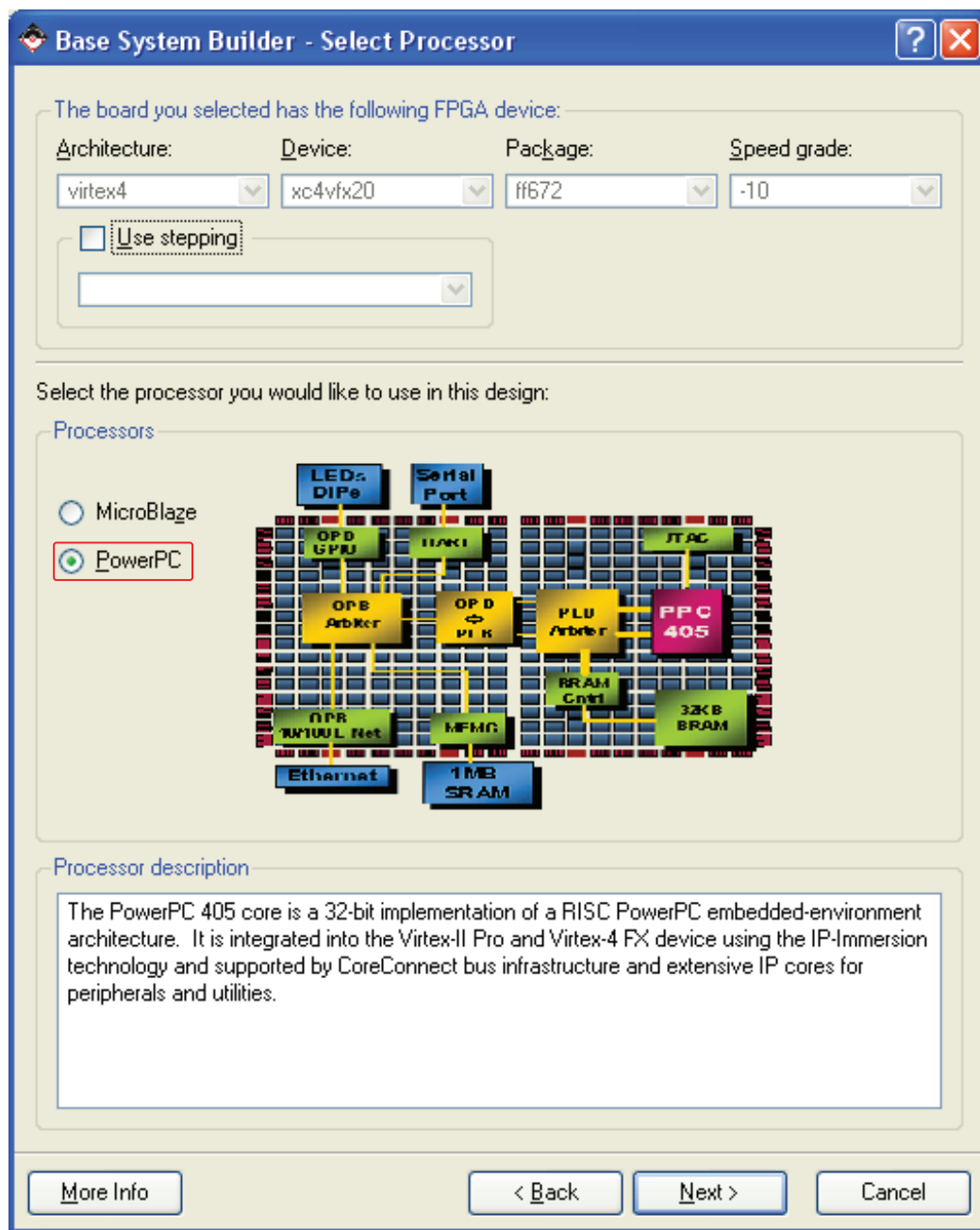


Figure 5: Selecting a Board in BSB

In the BSB - Select Processor window shown in Figure 6, PowerPC appears as the default processor because the design uses the Virtex™-4 FPGA architecture. Click **Next**.



X1023_06_080807

Figure 6: Selecting a Processor in BSB

In the BSB - Configure Power PC window shown in Figure 7, change the processor clock frequency to **300 Mhz**, then click **Next**.

Base System Builder - Configure PowerPC

PowerPC™

System wide settings

Reference clock frequency: 100.00 MHz

Processor clock frequency: 300.00 MHz

Bus clock frequency: 100.00 MHz

Ensure that your board is configured for the specified frequency.

Reset polarity: Active LOW

Processor configuration

Debug I/F

- ☒ FPGA JTAG
- ☐ CPU debug user pins only
- ☐ CPU debug and trace pins
- ☐ No debug

On-chip memory (DCM)
(Use BRAM)

Data: NONE

Instruction: NONE

Cache setup

☐ Enable

For optimal performance, enable burst and/or cacheline on memory

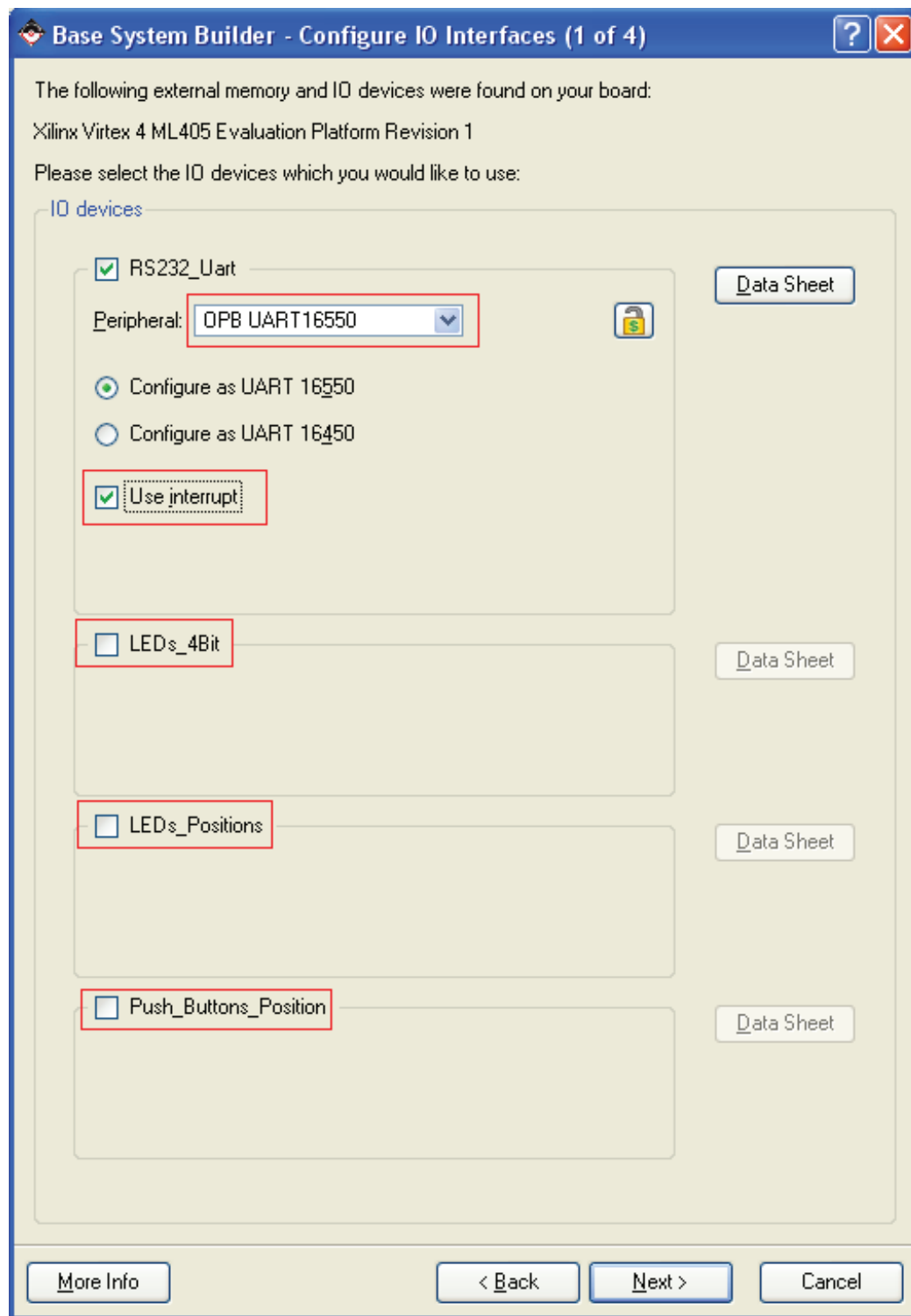
☐ Enable floating point unit (FPU) ?

More Info < Back Next > Cancel

X1023_07_091207

Figure 7: Setting the Processor Clock Frequency

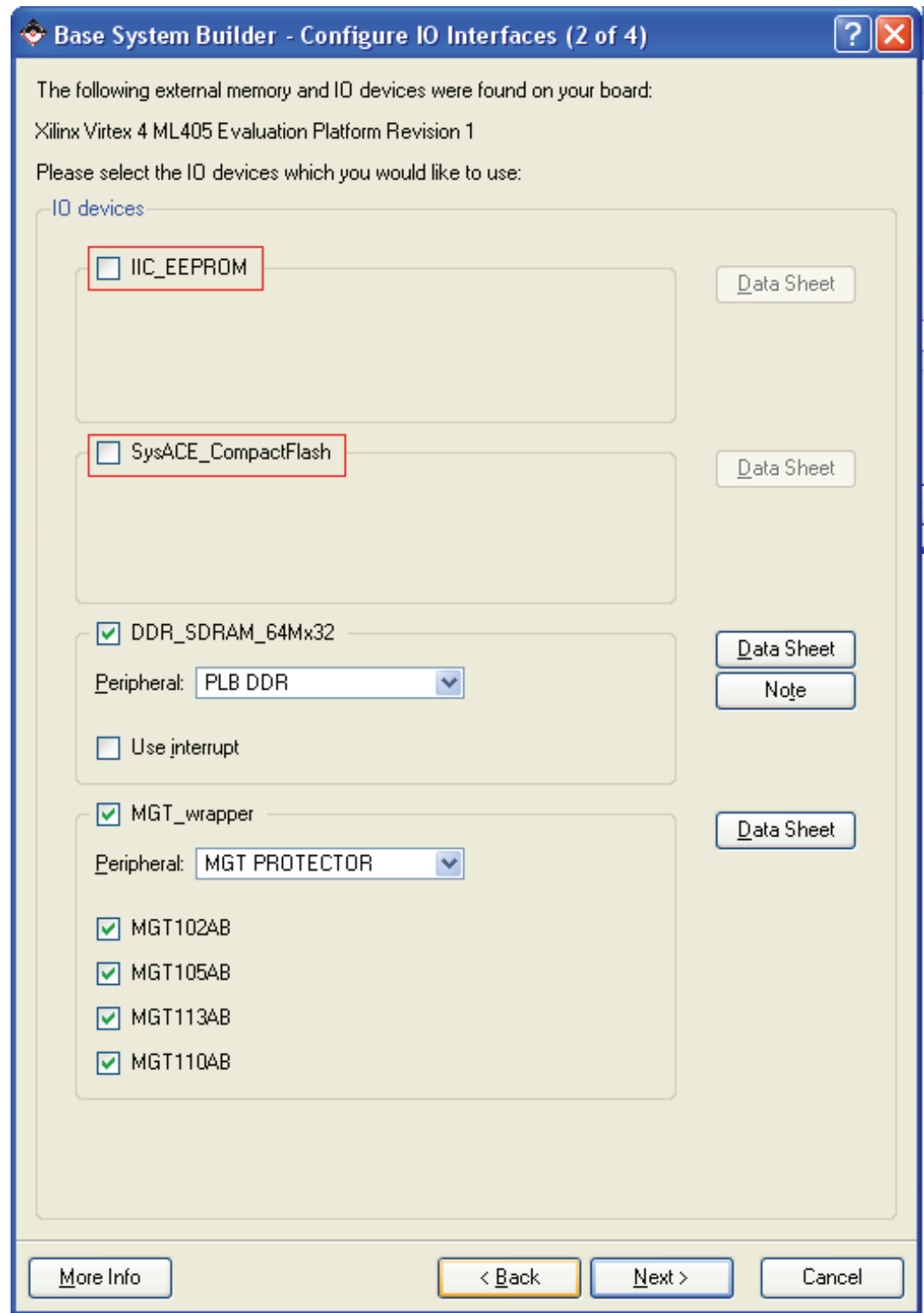
In the BSB - Configure IO Interfaces (1 of 4) window shown in [Figure 8](#), change the Peripheral to use the **OPB UART 16550** and check **Use Interrupt**. Deselect **LEDs_4Bit**, **LEDs_Positions**, and **Push_Buttons_Position**, as these are not required for this example.



X1023_08_091207

Figure 8: Selecting UART and Enabling Interrupts

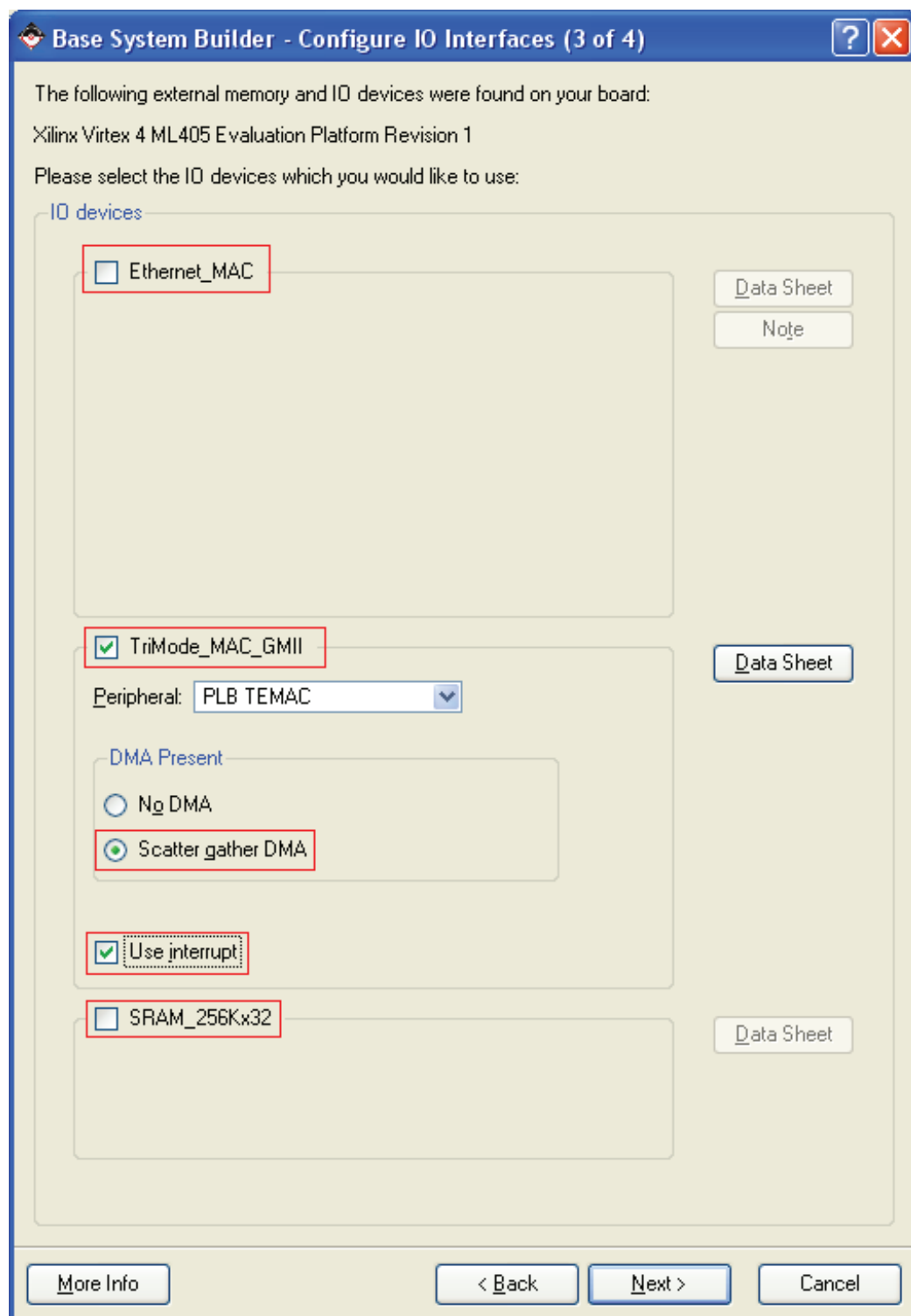
In the BSB - Configure IO Interfaces (2 of 4) window shown in Figure 9, deselect **SystemACE_CompactFlash** and **IIC_EEPROM**, because booting will be done off NFS which does not require IIC_EEPROM. Click **Next**.



X1023_09_09127

Figure 9: Selecting the IO Devices

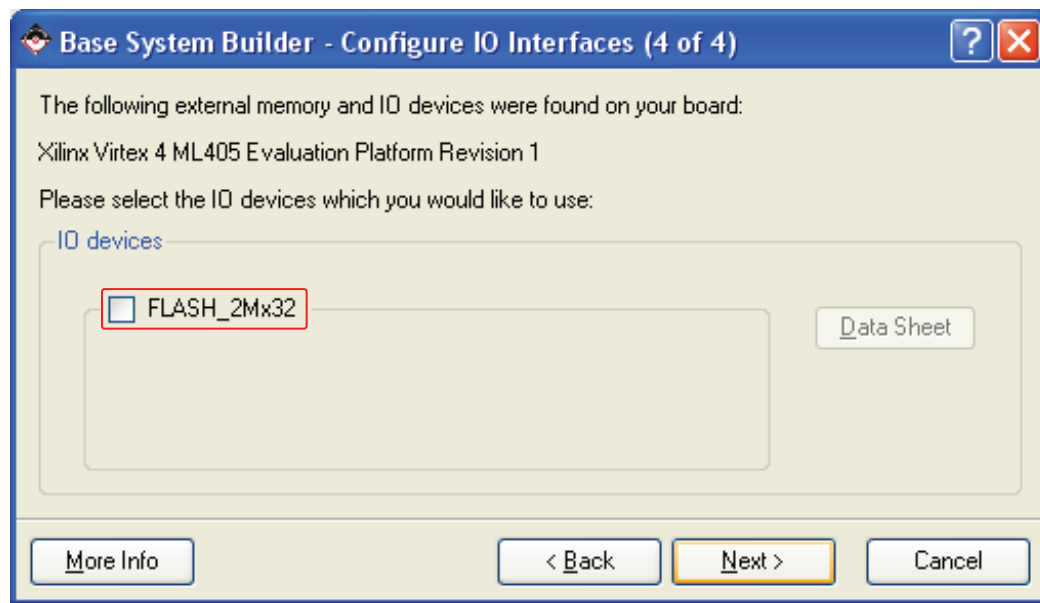
In the BSB - Configure IO Interfaces (3 of 4) window shown in [Figure 10](#), deselect **Ethernet_MAC** and select **TriMode_MAC_GMII**. Set the DMA Preset to **Scatter Gather DMA** and select **Use Interrupts**. Deselect **SRAM_256Kx32**.



X1023_10_091207

Figure 10: Selecting TriMode MAC GMII

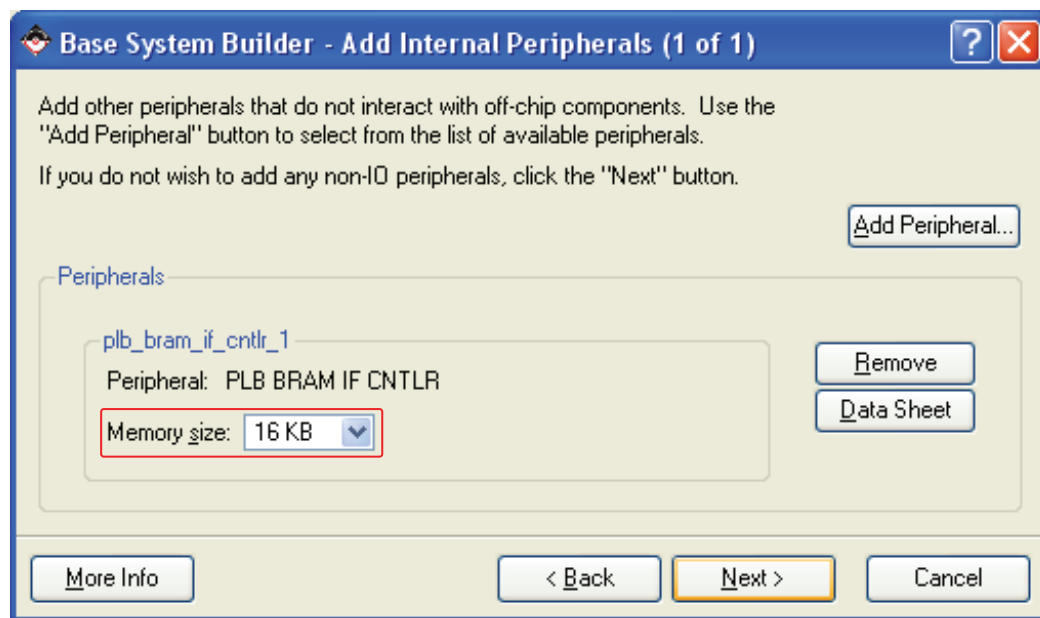
In the BSB - Configure IO Interfaces (4 of 4) window shown in [Figure 11](#), confirm that **Flash_2Mx32** is not selected, then click **Next**.



X1023_11_091207

Figure 11: Disabling Flash

The PLB Bram holds the boot vector of the PPC. In the BSB - Add Internal Peripherals window shown in [Figure 12](#), accept the Memory Size default value of **16KB**, then click **Next**.



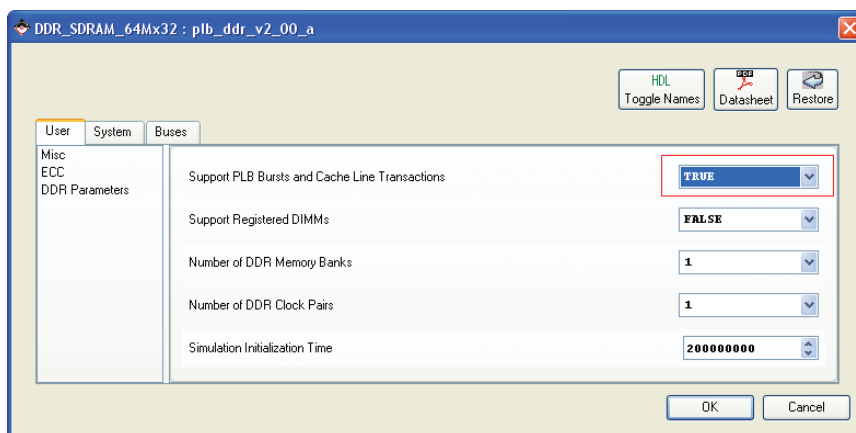
X1023_12_091207

Figure 12: Adding Internal Peripherals

Accept the default options for the test routines on the next few screens, by selecting **Next**. Finally, click on **Generate** to generate the processor system and **Finish** to exit the wizard.

Modifications to Improve Performance

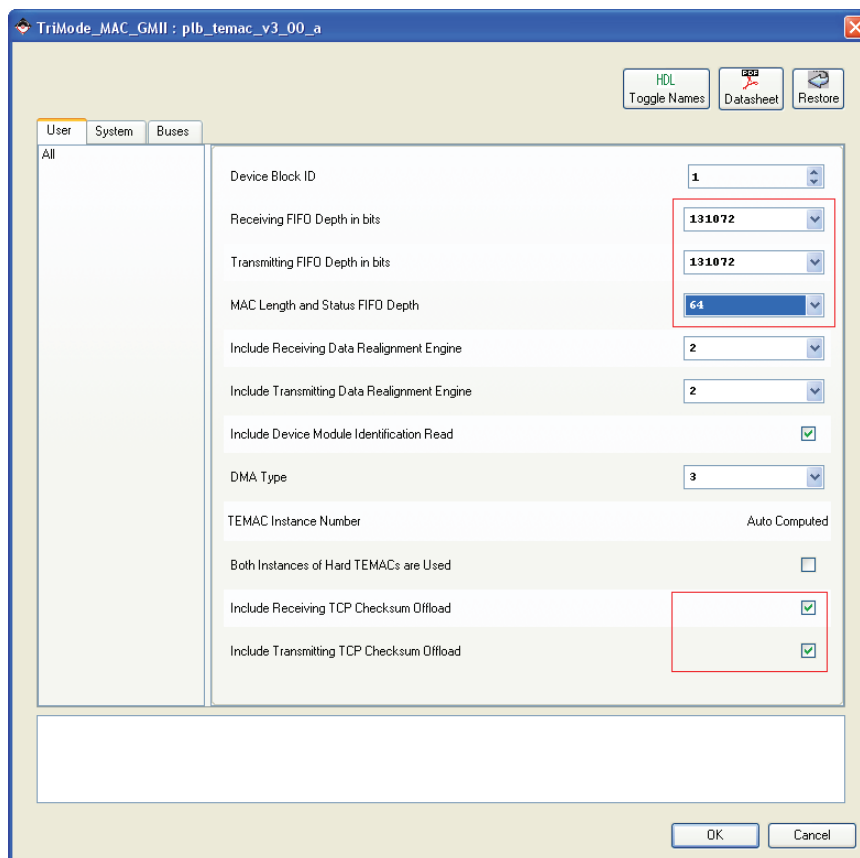
Right click on the system-assembly view of the DDR controller **DDR_SDRAM_64Mx32**. In the configuration window shown in Figure 13, select Support PLB Bursts and Cache Line Transactions to be **TRUE**, then click **OK** to save.



X1023_13_091207

Figure 13: Setting PLB Bursts and Cacheline Transactions

Right click on the PLB_Temac instance **TriMode_MAC_GMII**, make changes to the FIFO sizes depths and Checksum offload settings as shown in Figure 14, then click **OK**.



X1023_14_091207

Figure 14: Configuring the PLB TEMAC Instance

At this stage, the modifications to the hardware are complete. Select **Tools** → **Generate Bitstream** to generate the required system hardware. On completion of generation, a warning about licensing of the 16550 core may be displayed. If this is the case, the generated bitstream will be fully functional, however the uart will stop functioning after a number of hours.

Setup of a NFS Fileserver for the Linux Root File System

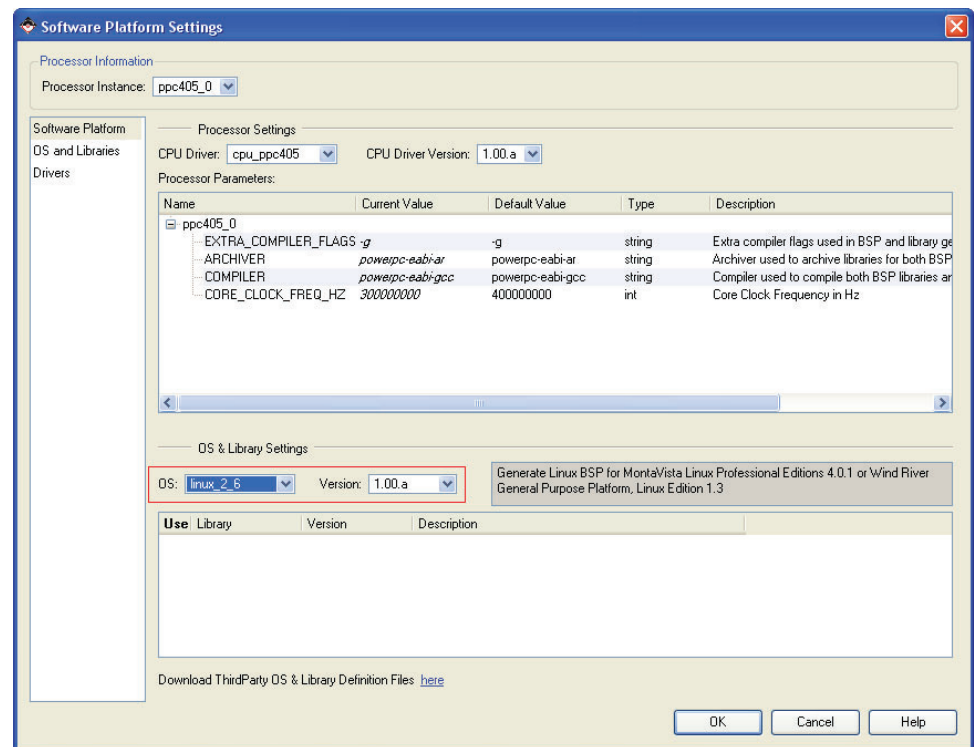
It is beyond the scope of this document to detail the specifics of how to setup a root file system on a linux server, mainly due to the instructions being different between linux distributions. Copy the contents of the MontaVista-provided target directory, `/opt/montavista/pro/devkit/ppc/405/target/`, to an nfs mount, with the following options in `/etc/exports`: `/nfsroot (rw,no_root_squash, no_all_squash, async)`

In the OS pulldown menu, select **Linux_2_6**, then click on the **OS and Libraries** option.

Kernel Configuration

In the XPS Software Platform Settings window shown in [Figure 15](#), select **Software** → **Software platform settings** to configure the required modifications to the Linux kernel.

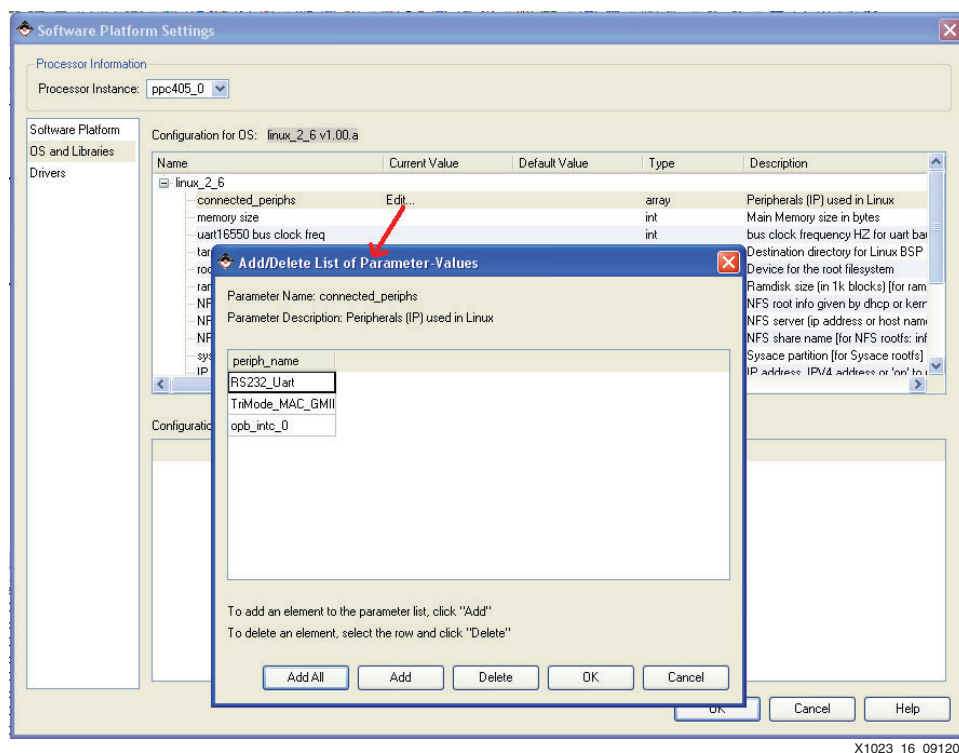
In the OS pulldown menu, select **Linux_2_6**, and then click on the **OS and Libraries** option.



X1023_15_091207

Figure 15: Configuring the Linux Kernel

In the XPS Software Platform Settings window shown in [Figure 16](#), click on **Edit** so that the connected_periphs connect the uart, temac, and interrupt controller to the linux BSP generation. Click **OK** to accept the default additions of the supported peripherals.

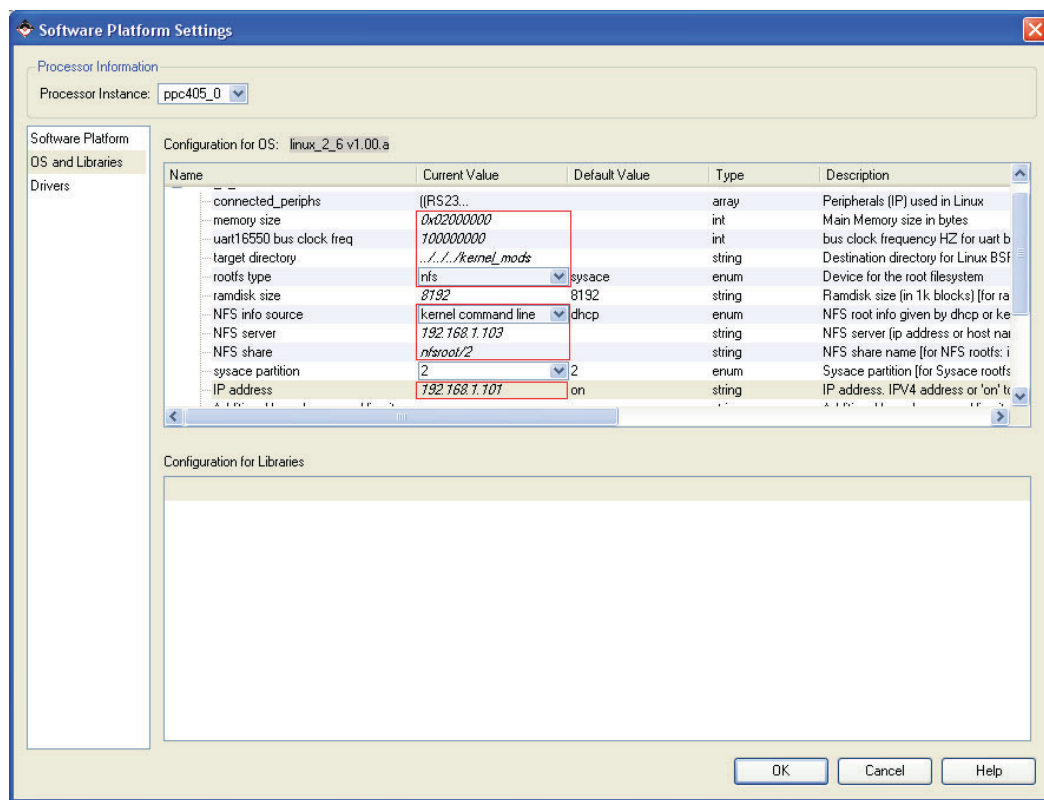


X1023_16_091207

Figure 16: Configuring the BSB Generation

Continue configuring the kernel patches by specifying the correct values for each field as shown in Figure 17. The memory size should be the same (or smaller) than the physical DDR on the board – 0x02000000. Set the speed of the bus to 100000000 (100Mhz). Set the target directory to ../../kernel_mods to make a patch directory in the root of the project. Set the NFS information source to be the kernel command line.

The nfs server entry has to match the IP address of the nfs server. Place the IP address in this field, along with the path to the network share in the NFS share field. For example, if the nfs share was 192.168.1.103:/nfsroot/2, enter 192.168.1.103 in the NFS server field, and nfsroot/2 in the NFS share field. If the ML405 is plugged into a DHCP network, leave the IP address set to on. If a CAT5e cable is used to connect the board to a server, specify an IP address in the same subnet as the server, for example, 192.168.1.101. Click **OK** to save the changes.



X1023_17_091207

Figure 17: Configuring the Kernel Patches

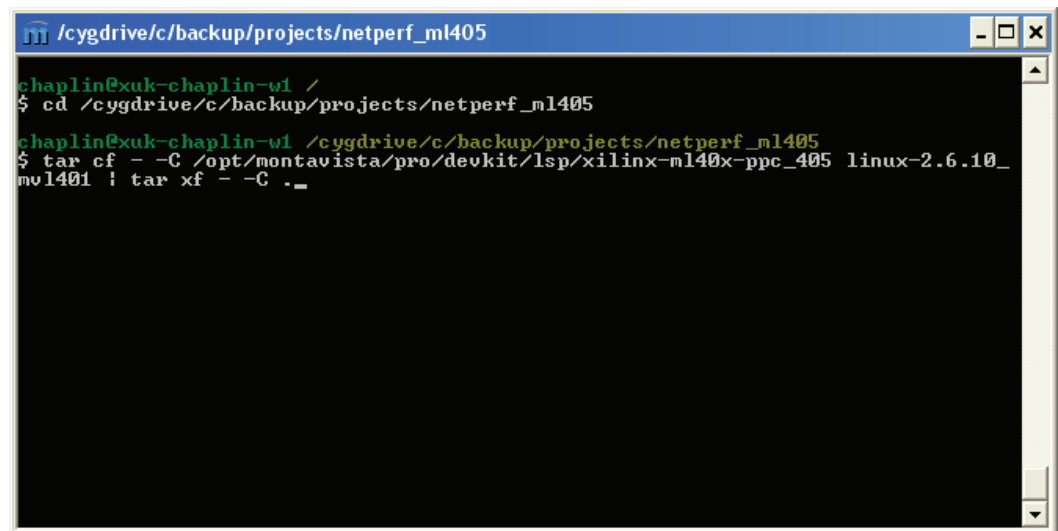
Generate the kernel patches by selecting **Software** → **Generate Libraries and BSPs** from the menu. A directory called kernel_mods (or whatever was specified in the target_directory field) will be generated with modifications to the default MontaVista Linux image.

Creating a Local Kernel Image for Patching

In MontaVista Linux, the kernel source tree is located within the installation. This source tree should not be modified directly, but rather be copied locally and modified there. Open a MontaVista Linux bash shell, change directory to the root directory of the project, then copy the kernel locally using the following command.

```
tar cf - -C /opt/montavista/pro/devkit/lsp/xilinx-m140x-ppc_405 linux-2.6.10_mv1401 | tar xf - -C
```

This command is shown in Figure 18.



X1023_18_091207

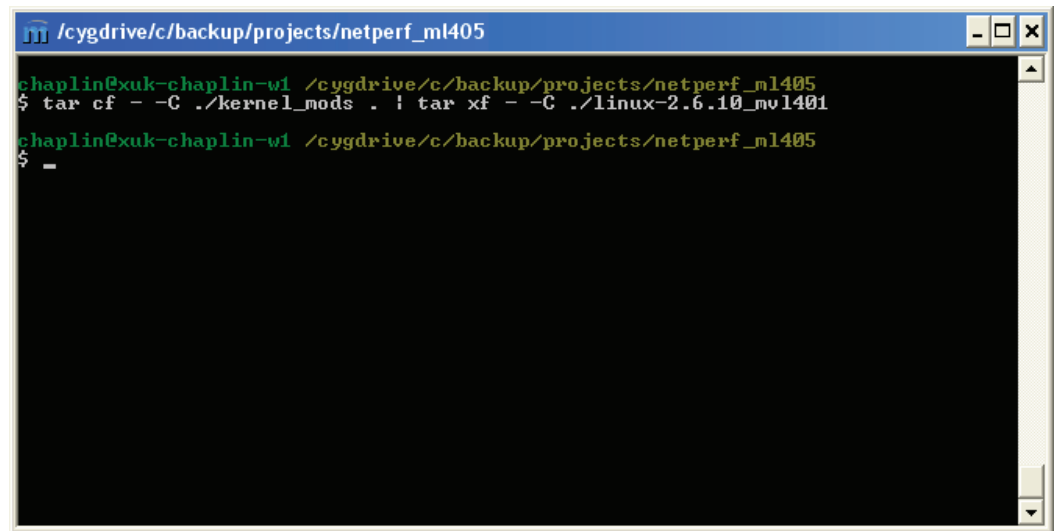
Figure 18: Creating the Local Kernel Image

This produces a copy of the linux kernel in a local directory `linux-2.6.10_mv1401`.

Patching the Kernel

It is possible to patch the kernel directly from XPS by specifying the kernel path in the software platform settings option **target_directory**. If that has been done, this step can be skipped. However, the user may prefer having a separate `kernel_mods` directory for more strict control of when the kernel files are patched. The contents of the `kernel_mods` directory must be copied on top of the kernel source files, overwriting the source where applicable.

These files and folders can be dragged-and-dropped in a windows platform or a command line approach can be taken as shown in [Figure 19](#).

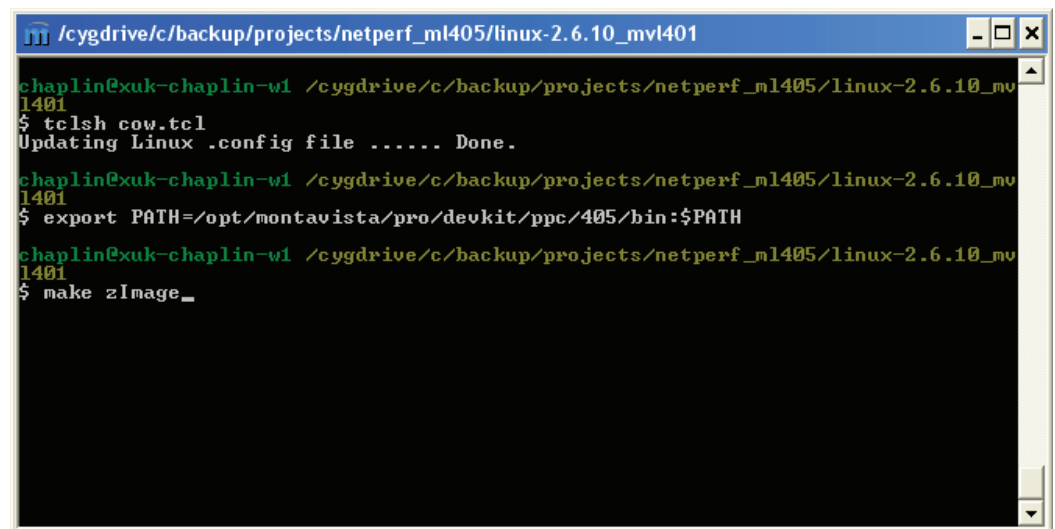
A terminal window titled "/cygdrive/c/backup/projects/netperf_ml405" showing a user named chaplin@xuk-chaplin-w1. The user enters the command "tar cf - -C ./kernel_mods . ; tar xf - -C ./linux-2.6.10_mvl401" to patch the kernel. The prompt returns to the user.

```
chaplinoxuk-chaplin-w1 /cygdrive/c/backup/projects/netperf_ml405
$ tar cf - -C ./kernel_mods . ; tar xf - -C ./linux-2.6.10_mvl401
chaplinoxuk-chaplin-w1 /cygdrive/c/backup/projects/netperf_ml405
$
```

X1023_19_091207

Figure 19: Patching the Linux Kernel

Following the kernel patching, the kernel can be configured automatically to use all of the supported, connected peripherals by running a generated `tcl` file. Execute this file by using the command, `tclsh cow.tcl`. After ensuring that the `ppc_405-gnu` tools are in the path, the kernel image can be built by executing `make zImage`. All of these commands are shown in [Figure 20](#).

A terminal window titled "/cygdrive/c/backup/projects/netperf_ml405/linux-2.6.10_mvl401" showing the same user. The user enters "tclsh cow.tcl", which outputs "Updating Linux .config file Done.". Then the user enters "export PATH=/opt/montavista/pro/devkit/ppc/405/bin:\$PATH". Finally, the user enters "make zImage_".

```
chaplinoxuk-chaplin-w1 /cygdrive/c/backup/projects/netperf_ml405/linux-2.6.10_mv
l401
$ tclsh cow.tcl
Updating Linux .config file ..... Done.
chaplinoxuk-chaplin-w1 /cygdrive/c/backup/projects/netperf_ml405/linux-2.6.10_mv
l401
$ export PATH=/opt/montavista/pro/devkit/ppc/405/bin:$PATH
chaplinoxuk-chaplin-w1 /cygdrive/c/backup/projects/netperf_ml405/linux-2.6.10_mv
l401
$ make zImage_
```

X1023_20_091207

Figure 20: Building the Kernel Image

On successful completion of the kernel build, the file `zImage.elf` will be available in the kernel directory, `linux-2.6.10_mvl401/arch/ppc/boot/images`.

Bitstream Download and Kernel Execution using XMD

Once the kernel and hardware have been successfully built, the initial hardware can be brought up for testing. Selecting **Device configuration** → **Download Bitstream** to download the bitstream to the ML405 board. The ML405 board should become configured, and the green *Done* light should be lit.

Select **Debug** → **Launch XMD** to launch the XMD command line debugger. If a configuration dialogue appears, accept the default options to launch the debugger. Upon a successful connection to the board, the debugger window should look as shown in [Figure 21](#).

```

c:\tools\EDK91\bin\nt\xmd.exe

Device  ID Code      IR Length  Part Name
-----
1       0a001093         8      System_ACE
2       05059093        16      XCF32P
3       01e64093        10      XC4UFx20
4       09608093         8      xc95144x1

PowerPC405 Processor Configuration
-----
Version.....0x20011470
User ID.....0x00000000
No of PC Breakpoints.....4
No of Read Addr/Data Watchpoints...1
No of Write Addr/Data Watchpoints...1
User Defined Address Map to access Special PowerPC Features using XMD:
  I-Cache <Data>.....0x70004000 - 0x70003fff
  I-Cache <TAG>.....0x70004000 - 0x70007fff
  D-Cache <Data>.....0x78004000 - 0x78003fff
  D-Cache <TAG>.....0x78004000 - 0x78007fff
  DCR.....0x78004000 - 0x78004fff
  TLB.....0x70004000 - 0x70007fff

Connected to "ppc" target. id = 0
Starting GDB server for "ppc" target <id = 0> at TCP port no 1234
XMD%
  
```

X1023_21_091207

Figure 21: Downloading the Bitsream and Executing the Kernel in XMD

Change directory to the location of the kernel image, and upload the image to the board. Type `con` to run the kernel, and observe the kernel output in a suitable RS232 terminal (9600 baud, hardware handshake, 8n1). If using the MontaVista-provided target filesystem, login using the user `root` with no password.

Sourcing and Compilation of netperf

Netperf is available as a source tar file from www.netperf.org. Download the most current version of the tool. Make the `tar.gz` file available in the file system of the target board for target compilation.

Before extracting any files, ensure that the date is set on the target. Because there is no battery-backed up date available, this can be done manually. See [Figure 22](#).

```

root@192.168.1.101:~# date -u 062515532007
Mon Jun 25 15:53:00 UTC 2007
root@192.168.1.101:~# █
  
```

X1023_22_080807

Figure 22: Sourcing the Compiling netperf

Extract the files using `tar`, and change to the directory created.

Configure the build environment by executing `./configure`. This should result in many checks, as well as the generation of a custom make environment for the target.

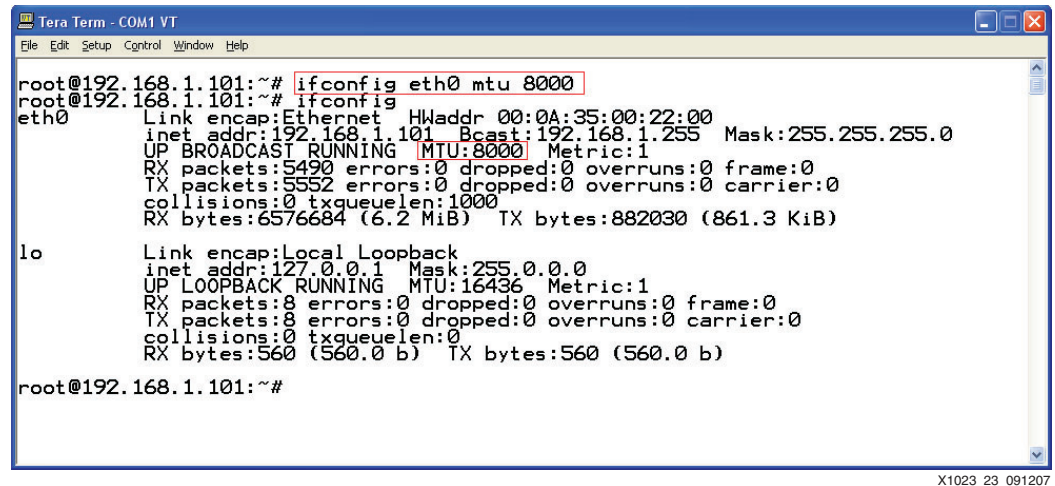
The netperf tools can then be compiled by using the command `make`. Install the tools by typing `make install`.

Configuration of Network Interfaces to Support Jumbo Frames

To get the best performance from the Ethernet interfaces, the Maximum Transmission Unit (MTU) of the interface can be set to a size to support Jumbo Frames. This setting is required on both the server as well as on the client to enable optimal packet sizes.

On both the client and the server, use `ifconfig` (as root) to change the MTU to **8000**.

Figure 23 shows this for the ML405.



```

Tera Term - COM1 VT
File Edit Setup Control Window Help
root@192.168.1.101:~# ifconfig eth0 mtu 8000
root@192.168.1.101:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0A:35:00:22:00
          inet addr:192.168.1.101  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MTU:8000  Metric:1
          RX packets:5490 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5552 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6576684 (6.2 MiB)  TX bytes:882030 (861.3 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:560 (560.0 b)  TX bytes:560 (560.0 b)

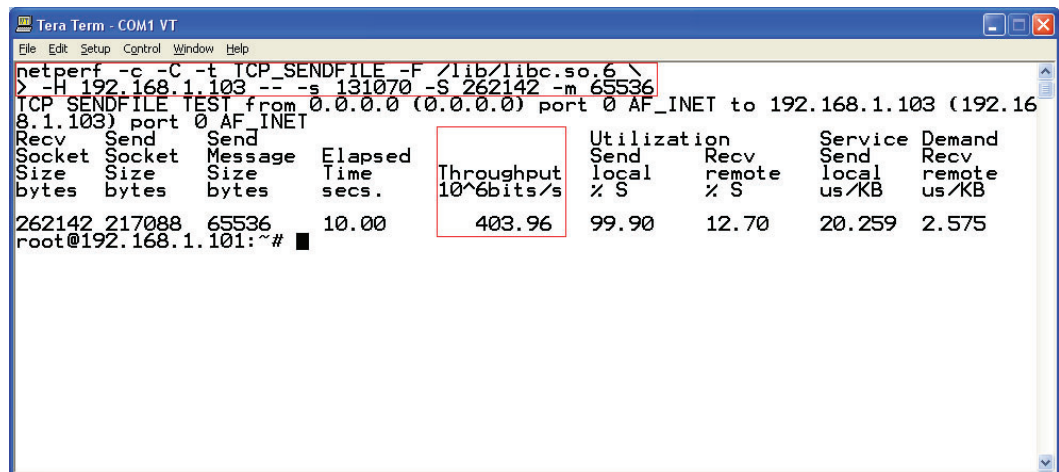
root@192.168.1.101:~#

```

X1023_23_091207

Figure 23: Configuring Interfaces to Support Jumbo Frames

netperf commands can then be run from the console. This requires netserver to be running on the target server. See the netperf readme files for more information.



```

Tera Term - COM1 VT
File Edit Setup Control Window Help
netperf -c -C -t TCP_SENDFILE -F /lib/libc.so.6 -H 192.168.1.103 -- -s 131070 -S 262142 -m 65536
TCP_SENDFILE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to 192.168.1.103 (192.16
8.1.103) port 0 AF_INET
Recv  Send  Send
Socket Socket Message Elapsed Throughput Utilization Service Demand
Size Size Size Time 10^6bits/s % S % S local Recv local Recv
bytes bytes bytes secs. 10^6bits/s % S % S us/KB us/KB
262142 217088 65536 10.00 403.96 99.90 12.70 20.259 2.575
root@192.168.1.101:~#

```

X1023_24_091207

Figure 24: Running netperf Commands

As shown in Figure 24, a performance of over 400 Mbits per second is achievable for an unmodified kernel.

Kernel Modifications to Further Improve Performance

The default kernel files set the TX and RX thresholds to values of **16** and **2**, respectively. The frequency of interrupts can be controlled with the interrupt coalescing features of the SG DMA engine within the TEMAC core. These features can be used to optimize interrupt latency and throughput for the user's network traffic conditions. The packet threshold count will delay

processor interrupts until a programmable number of packets have arrived or have been transmitted. The packet wait bound timer can be used to cause a processor interrupt even though the packet threshold has not been reached. The timer begins counting after the last packet is processed. If no other packet is processed as the timer expires, then an interrupt will be generated.

The values of the TX and RX thresholds can be found in the kernel file `drivers/net/Xilinx_temac/adapter.c`. The file can be edited before re-compiling the kernel. With values of **32** and **8** for TX and RX threshold, the performance can increase as illustrated below in [Figure 25](#).

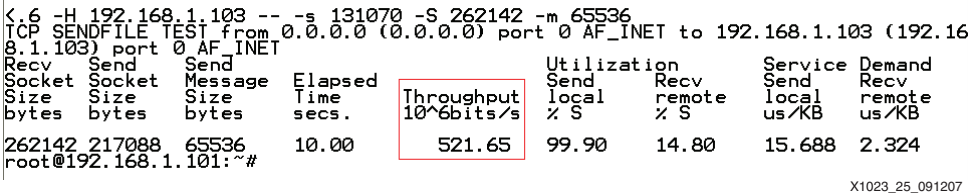


Figure 25: Editing Threshold Values

References

UG410 (v1.0.1) June 30, 2006 ML405 Evaluation Platform User Guide

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/3/07	1.0	Initial Xilinx release.